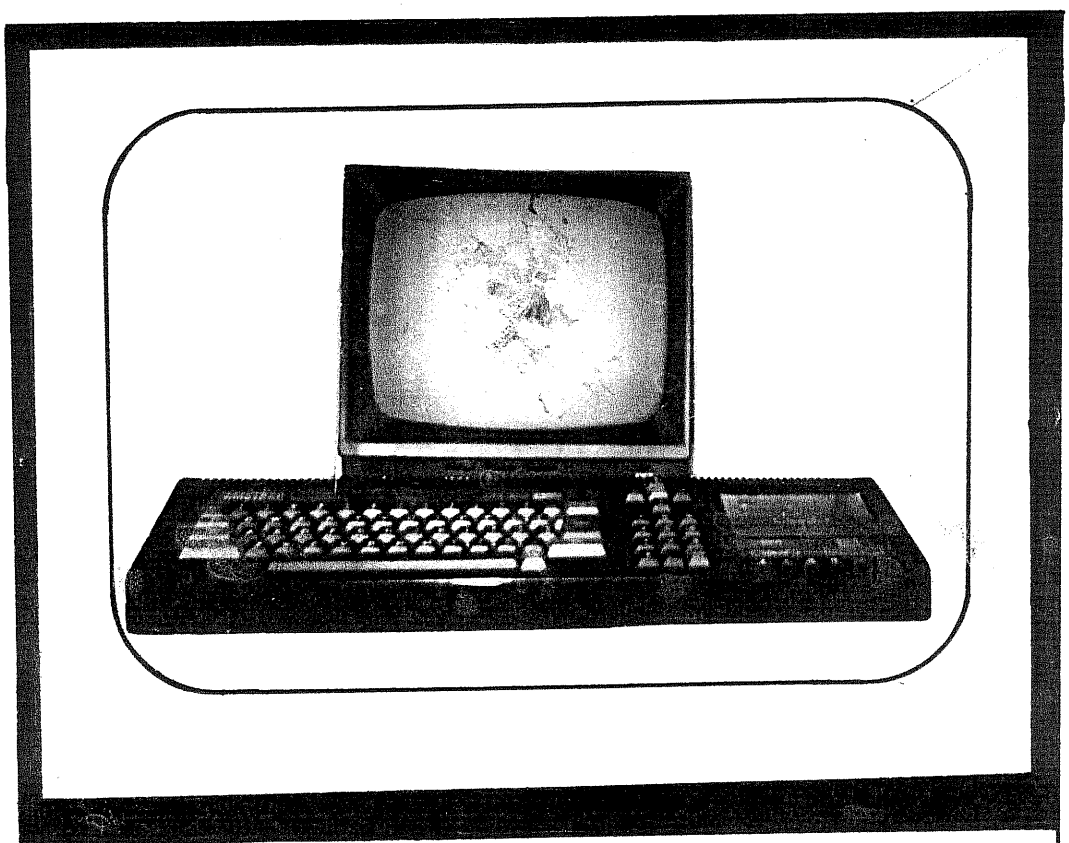
AMSTRAD
PRINT—OUT

Issue One

Price 70p



BY Thomas Defoe, Mark Gearing and Jonathan Haddock.

What is my Amstrad ? - PART 1

....also including....

LANGUAGE TESTER - Mammoth listing

WHICH JOYSTICK? - Review

BASIC & Machine code - Part one

ADVENTURES - Explored

HARDWARE PROJECT - Build-it-yourself add-on

INDEX – ISSUE ONE

Miscellaneous

Features

Reviews

Programming

We would like to take this opportunity to thank Mr Gearing for the use of his photocopier in the production of this magazine and also to state that we do not support piracy in any form whatsoever.

This is the first issue and so you are unlikely to be familiar with all the items we wish to bring to you. We will rely, to a certain extent, on items that you send us and so we would be grateful if you could send us any of the items shown below.

PROGRAMS - We will always need programs of any sort and in any language as we try to provide variety and interest. If you have a program and would like to see it published in future issues, please send it to us at the address shown below. If you wish your work to be returned please send a suitable SAE.

LETTERS - We are always grateful for any letters that you send us whether they are scathing or praising the magazine. In future, we will try and publish as many as possible and you can write to us about any CPC related matter at the address shown below.

USER CLUBS - If you run or are a member of any User Club throughout the world please send us the address of the Club and information on any fees and we will include you in the regular User Club Contacts section.

HOMEBREW SOFTWARE - If you are selling a program for the Amstrad send it to us and we will be very pleased to review it and include it in our next publication.

10 LINE PROGRAMS - As the name suggests you can submit any program in any language as long as it is no more than TEN lines long. Every month there will be a small competition and the prize will go to the author of what we consider is the best program.

SMALL ADS. - If you want to sell something, just send us your advertisment of no more than 20 words (excluding address and telephone number) and we will print it in the next issue of the magazine absolutely free of charge.

The address to send any of these to is :-

Thomas Defoe,
8, Maze Green Road,
Bishop's Stortford,
Hertfordshire.
CM23 2PJ.

Adventuring : past, present & future....

HAVE you ever entered myths and legends to save a beautiful princess from a terrible catastrophe, or defeated Chicago gangsters who are plotting to put deadly drugs on the streets ? All this is possible with adventure games. If you have, at any time, played an adventure game, you will already know how exciting and also compulsive they can be.

MAYBE these games do not provide a realistic image in themselves, in fact it is probably the fantasy element which is the key to their success, but when combined with stunning graphics, detailed descriptions, an intriguing plot and a fair bit of imagination, even the most critical gamesplayers could find them enjoyable. Even so, the make or break point of the game is whether you have the patience to solve the puzzles. Some people dislike the thought of wandering around foreign lands and casting spells - this is no excuse ! There are plenty of adventure games set in our times and dealing with real life, if uncommon, situations. An excellent example is CORRUPTION which involves insider trading and detective work in the 1980s. The only category of people who have a genuine excuse are those who find the games and puzzles they involve far too cerebal.

SINCE the very first Colossal Cave adventure there has been dramatic progress. Originally the parser (the way the computer understands your instructions) was only able to understand very simple commands such as - 'OPEN DOOR' or 'GO NORTH' - but recently companies such as Magnetic Scrolls and Level 9 have been steadily increasing the size and capabilities of their parsers until sentences such as 'GO SOUTH, WAIT 1, LOCK DOOR WITH RUSTY KEY, KILL VAMPIRE' - are not unheard of. This is not the only improvement as very detailed graphics are now standard and go with nearly every piece of text. Looking to the future, we shall soon see (or rather hear!) digitized speech as we meet the interactive characters who lurk in dungeons. With the introduction of other characters, the games have become less and less predictable from the simple, go somewhere, solve a puzzle, go somewhere else.

TWO other branches of adventuring have emerged over the past few years. These are the 'Homebrew' type of adventures which have often been produced using The Quill, GAC or some other adventure writing program. Some of these have been entertaining and even commercially viable products although the majority are not great successes.

GAMES which are based on role-playing have also emerged and require great skill, tactics and planning. In them you play the part of two or more characters who collaborate with each other and assisting themselves in their quests. They still have all of the adventure game characteristics, such as plots, puzzles and objects, and yet are not true adventure games. They are more of a strategy game, or even a simulation, set in another world or time but are still classed, by many, as adventure games. There are at the moment, for the Amstrad, very few of them, but perhaps the best is the Bard's Tale.

IN this exciting role-playing game, which is set in the town of Skara Brae, you control a group of six characters. Your group must explore the town and its secrets in an attempt to destroy the evil mage, Mangar. For Mangar has frozen the surrounding lands and so prevents outside help from reaching the town. Evil creatures have entered the town and it is left to your group to destroy Mangar and to save Skara Brae.

You can choose your group from a huge range of characters as there are seven races (human, elf, dwarf, hobbit, half-elf, half-orc and gnome) who can each be one of eight professions. As well as this they have five different attributes, each of which ranges from 1 to 18.

There are four methods of attack, by weapon, by spell, by magic item or by singing a Bard Song. After defeating monsters items can sometimes be picked up by the party. The Bard's Tale also has a complicated experience system whereby your characters can gain skill levels according to the number of monsters that they kill. Mages also have a large number of spells available to them and so can become very powerful.

Within the town there is a shop, a magic emporium, two towers, taverns, temples and the Review board as well as the Adventurer's Guild and shops. The Review board is the place where characters can increase their skill so long as they have enough experience. They can also increase a mage's spell capacity,but for a price !! Not only is there the town to explore but there are cellars, catacombs, sewers and two towers to explore and they ensure the game's addictiveness. 93%
All in all, at £8.95 (cass.) or £14.95 (disc) it is very good value for money.

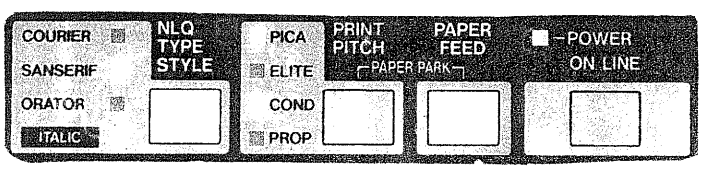
ONE bad point for humble CPC464 owners, like myself, is that more and more adventure games are disc only and this limits our choice in the very thin market even more. For a list of games see the 'Games Reviews' section of the magazine.

Printers

LC 10

The LC-10 is a 9 pin dot matrix printer and has 8 different fonts and four types of spacing. In draft mode it prints 144 characters per second and in near letter quality (NLQ) mode 36 cps. It has a 4K text buffer and also features paper-parking. This means that you can use single sheets of paper without removing the continuous stationary. You can also buy the LC-10 Colour which enables you to print text in black, red, green, blue, purple, yellow and pink. All functions are easy to control and adjust via a simple front panel and so there is no more fiddly DIP switches. The black and white version now costs about £229 and the colour version about £269. You can contact Star at :- Star Micronics UK Ltd., Craven House, 40 Uxbridge Road, Ealing, London W5 2BS, or by telephone on :- 01 8401829. With all the advantages of colour, paper-parking and the front control panel the Star LC-10 represents excellent value for money. If you are looking for a printer - get this ! Highly recommended.

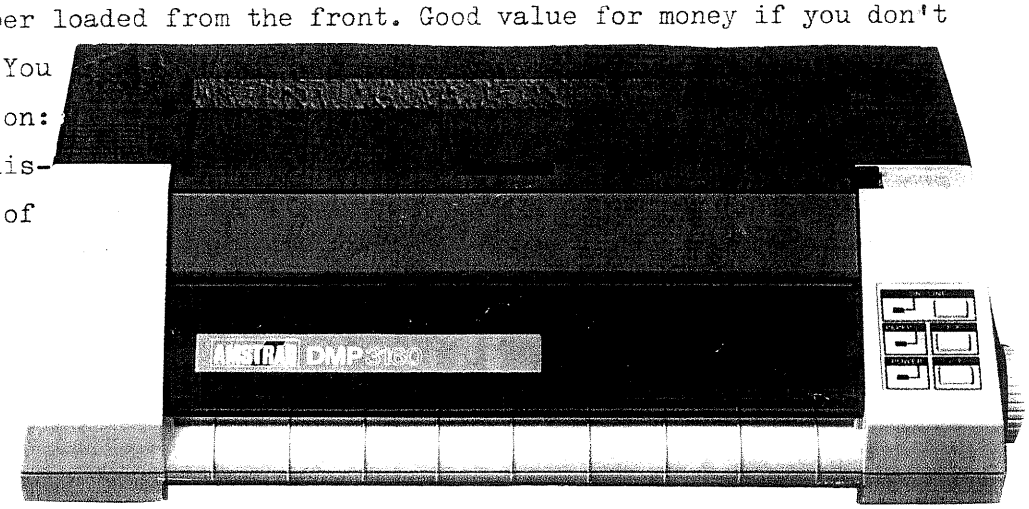
90%



DMP 3160

After the LC-10, this printer seems a very normal one. It costs £228.85 and is usable with any computer which has a standard Centronics parallel interface. It only prints in black and white but is faster at printing than the LC-10, having a print speed of 160 cps in draft mode and 40 cps in NLQ mode. Unlike most printers where paper is loaded from the back, the DMP 3160 and its little brother, DMP 2160, have to have their paper loaded from the front. Good value for money if you don't want to print colour. You can telephone Amstrad on: 0277 230222. The one disadvantage is the cost of ribbons; £5.99 each.

78 %



WHAT IS MY AMSTRAD ?

An Introduction to some of the Fundamental Principles used in the Computing Science.

For a newcomer, computing is a very complicated subject. It is full of technical jargon and all too often the manual doesn't answer your questions. In this section I shall try to explain the fundamental principles used in the science of computing without being too technical. I will also try to answer any queries that you have.

GLOSSARY - Before I can start to explain about your Amstrad you will need to know a few simple terms. Don't worry if you don't understand them fully at the moment as I will elaborate on them all later.

KEYBOARD - This is the part of the computer which looks a bit like a typewriter. It is used to give information to the computer.

MONITOR - This is the TV screen and is used to give you a visual representation of what the computer is doing.

HARDWARE - This is the actual machinery of the computer. eg. the keyboard.

SOFTWARE - The software is a PROGRAM (NB: Different spelling from a TV programme).

PROGRAM - A program is a set of instructions which tells the computer what to do.

FIRMWARE - This is a cross between software and hardware in that a program is stored permanently as part of the computer.

DISC DRIVE or TAPE RECORDER - A tape recorder or disc drive is used to store a program on a tape or disc indefinitely for use later. (Note: On the CPC 464 the tape recorder is called the 'Datacorder'.

EXECUTE or RUN - When the computer executes or runs a program it actually does what the program tells it to.

BASIC - When you switch on your Amstrad CPC (CPC stands for Colour Personal Computer you are faced with a copyright message and various bits of information about your computer. In this text you will see the word BASIC followed by a number. This number shows what version of BASIC you have. BASIC stands for Beginner's All-purpose Symbolic Instruction Code (so now you know why it is abbreviated!) and is a COMPUTER LANGUAGE. A computer language is a program which allows you to enter instructions from the keyboard and converts them into instructions (in the form of numbers) which the computer can understand and execute. On the Amstrad CPC, BASIC is part of the firmware and is called a RESIDENT language which means that it is built in. BASIC is one of the most common and popular languages because it is very simple to use. This is because most of its instructions (often called COMMANDS) are very similar to English words. Its only disadvantage is that it is quite slow when you are writing long programs.

MEMORY - Memory is (or can be) very complicated and yet it is very useful. When you switch on your Amstrad you are told how much memory you have available on the particular computer. On a CPC 464 or 664 you should see 64K somewhere amongst the text and this tells you the amount of memory you have. The K stands for Kilobytes and so you have 64 Kilobytes of memory available. You may have noticed that the amount of memory is represented in the number of your CPC. Both the 464 and 664 have 64 Kilobytes whilst the 6128 has 128 Kilobytes of memory. Unfortunately that was the simple part of memory and next we must deal with BINARY numbers. Although it is not completely necessary to understand memory and binary it is very helpful as the whole of the computing science ties in with it.

BINARY - As I mentioned above, you are going to need to be able to convert decimal numbers into binary. As I am not going to devote pages and pages to this subject it would be useful if you knew how to do this or had a calculator that was able to do it. Failing that, you could always use your trusty Amstrad to do it for you. The small program below also gives HEXADECIMAL numbers (more on them later).

Type in each line exactly as shown, remembering to press the ENTER key at the end of each line. When it has been typed in, type RUN followed by the ENTER key. It will then prompt you to enter a number between Ø and 255 followed by the ENTER key. The program will now print the number in decimal, hexadecimal and binary. To use it again type RUN followed by the ENTER key and repeat the instructions.

NUMBER CONVERTER PROGRAM -

```
10 REM Number Converter (c) 1989. T.J.Defoe
20 MODE 2
30 INPUT "Input a number from 0 to 255";a
40 PRINT TAB(3)"Decimal=" a;
50 PRINT TAB(20)"Hexadecimal= &"+HEX$(a);
60 PRINT TAB(41)"Binary= "+BIN$(a)
```

Note: It only converts decimal numbers and not vice versa.

BITS and BYTES - Computers only understand Øs and 1s (that is BINARY). Well, you may be asking, how come it understood the instruction 'MODE 2' in the above program? The answer is that the instruction 'MODE 2' (a BASIC command) was converted by BASIC into Øs and 1s for the computer to understand and EXECUTE when you typed in 'RUN'.

In a world where only Øs and 1s exist, how do you count past 1? In BASE 2 (BINARY) you have column headings like you do in BASE 10 (DENARY) but with different headings.

In Base 10 the column headings are all, ten times the previous column heading. So you have 1000s, 100s, 10s and units like so.

	1000	100	10	1	
eg.	6	4	7	2	= (6x1000)+(4x100)+(7x10)+(2x1) = 6472

However, in Base 2, the column headings are all 2 times the previous column heading.

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	
128	64	32	16	8	4	2	1	(in decimal)
1	Ø	Ø	Ø	1	Ø	1	Ø	= (1x128)+(1x8)+(1x2) = 138

Now if we add all the column headings up we get 255 (128+64+32+16+8+4+2+1). If you enter 255 into your calculator or use the number converter program you will see that -

255 (decimal) = &FF (hexadecimal) = 11111111 (binary)

Let's take the binary (11111111), each number is a BIT (which stands for Binary digIT A BIT shows us whether there is a certain number in a column. If there is,it is set to 1,if not,it is set to Ø. In other words it is a TRUE or FALSE indicator.

Your Amstrad handles 8 BITs at a time and 8 BITs are called a BYTE (a programmer's joke!). Therefore using one BYTE we have 256 possible variations; ØØØØØØØØ (Ø) to 11111111 (255). As 256 is not very big your Amstrad uses two BYTEs to form an ARRAY. The first byte gives the row and the second byte the column. (A BIT like computer battleships!)

	0	1	2	3	4	5	6
0							
1							
2				1			
3							
4							

The diagram above shows a small array. At BIT 2,3 a 1 is stored. Another word for any of the positions in the array is a Memory Location. Therefore, a binary array of size 256x256 can have 65,536 individual MEMORY LOCATIONS. So now Øs and 1s can identi any one of 65,536 locations. As I mentioned earlier you have Kilobytes (K) which are 1Ø24 bytes (the nearest multiple of 256 to 1ØØØ). If you divide 65,536 bytes by 1Ø24 bytes you get 64 Kilobytes which happens to be the number of memory locations that your computer tells you it has when you switch it on. So the number of memory location is the same as your computer's memory size. If you have a 6128, you have two sets of 64 memory which you can swap between at will very easily and quickly. However you can onl operate on one set at a time.

However, although Binary is (hopefully!) quite simple and straightforward it is longwinded and can often be entered wrongly. So to shorten the binary number we use hexadecimal (sometimes shortened to HEX). Hex is Base 16 and is like this:-

Decimal :	1	2	3	4	5	6	7	8	9	1Ø	11	12	13	14	15
Hexadecimal :	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Thus, if you take a Binary number eg. 11Ø1ØØ1Ø and split it into two groups of four bits (called a NIBBLE: no, I am serious!)

eg.

11Ø1¦ØØ1Ø

LOOK at the first group of four bits and convert them into decimal.

eg.

8	4	2	1	
1	1	Ø	1	= (1x8)+(1x4)+(1x1) = 13 (decimal)

And then convert that into Hexadecimal.

eg.

13 (decimal) = D (hexadecimal)

You then repeat the process with the second nibble.

eg.

8	4	2	1	
Ø	Ø	1	Ø	= (1x2) = 2 (decimal) = 2 (hexadecimal)

And then put them together,you find that :- 11Ø1ØØ1Ø (binary) = D2 (hexadecimal)

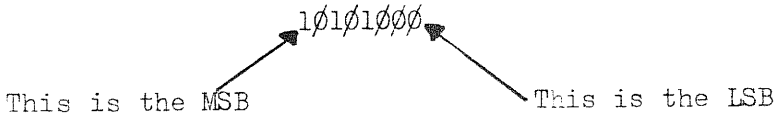
To convert back:- D2 =(13 x 16) + (2 x 1) = 21Ø (decimal)

However it is common practice to preceed a hexadecimal number by an & sign. eg. &D2

(Of course you could always use the Number Converter Program instead).

MOST and LEAST SIGNIFICANT BITS - If you take a binary number. eg. 1Ø1Ø1ØØØ in this the first digit is the Most Significant Bit (MSB) and the last digit is the Least Significant Bit (LSB).

eg.



Congratulations! If you've read all of this section and have managed to understand it all you are well on your way to knowing all about the number systems used in computing and should know quite a few technical terms and jargon. In the programs section there is a program which will test you on decimal, binary and hexadecimal. Now all you've got to learn is :- 2's Complement, Signed numbers....,

---

Program 1

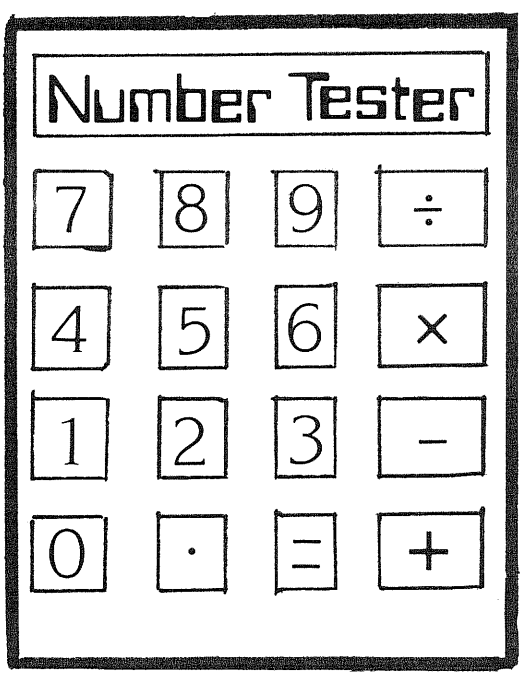
NUMBER TESTER 1

Now that you have read everything about Binary, Decimal and Hexadecimal in the 'What is my Amstrad ?' article, I am sure that you would love to test your new-found knowledge. Well now you can ! This program asks you for the total number of questions you wish to have asked, and then gives you a decimal number and you must convert it into both HEXADECIMAL and BINARY. It will check your answers to both and at the end will print your score and ask if you want another go.

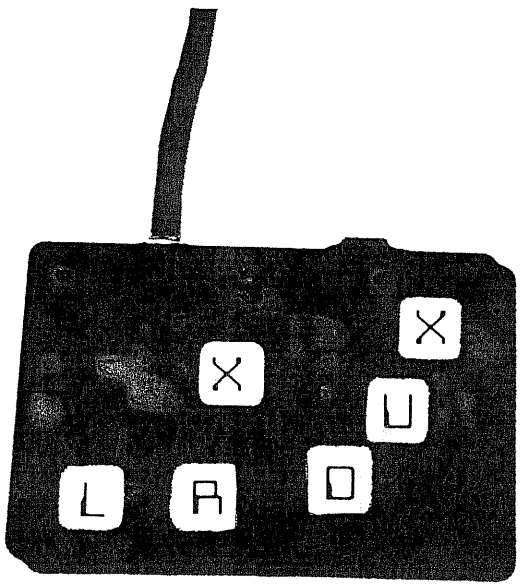
```
 10 REM Number Tester, (c) 1989 T. Defoe
 20 score=0
 30 MODE 2
 40 PRINT "How many questions do you want (1-100)";
 50 INPUT a
 60 IF a<1 OR a>100 THEN SOUND 1,100,5:GOTO 30
 70 FOR i=1 TO a
 80 b=INT(RND*255)
 90 PRINT:PRINT "What is";b;"in HEXADECIMAL";
100 INPUT c$
110 c$=UPPER$(c$)
120 d$=HEX$(b)
130 IF d$=c$ THEN PRINT "Correct":score=score+1
140 IF d$<>c$ THEN PRINT "Wrong":PRINT "It is ";d$
150 PRINT:PRINT "What is";b;"in BINARY";
160 INPUT c$
170 c$=UPPER$(c$)
180 d$=BIN$(b)
190 IF d$=c$ THEN PRINT "Correct":score=score+1
200 IF d$<>c$ THEN PRINT "Wrong":PRINT "It is ";d$
210 FOR t=1 TO 2000:next t
220 CLS
230 NEXT i
240 PRINT "You scored";score;"and got";(2*a)-score;"wrong"
250 PRINT "Do you want another go (y/n)";
260 INPUT z$
270 z$=UPPER$(z$)
280 IF LEFT$(z$,1)="Y" THEN RUN
290 END
```

NB. The range of numbers that you can be asked is, at present, 0 to 255. If you want to have lower or higher numbers, just change the value in line 8Ø. eg.

8Ø b=INT(RND*?) where ? is the new value.



JOYBOARD — easy to make, simple to use.



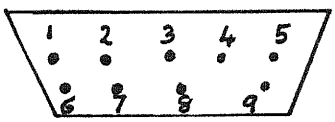
Introduction —

In this, the first part of what we hope will be a regular feature, we aim to give you some simple, cheap and FUN projects that you can do and make without any specialist equipment or using too much time. What you will need, however, is a normal tool set (screwdriver, pliers, hammer, etc.), a drill and a soldering iron (complete with solder). For this first project you will also require the use of either a hair-drier or an oven to provide heat. Most of our projects will cost under £10 and this is no exception - it costs £9.20 - but considerable savings can be made if you leave out the non-essential (*) parts.

Construction

First of all, you must feed the ribbon cable through the heat shrinkable tubing so that there is an inch of cable sticking out of each end. A good method of doing this, is to tie a piece of cotton around the cable and pull it through the tubing. Once you have done this, heat the tubing either using an oven or a hair-drier. It only starts shrinking when heated to over 120°C and is self-extinguishing but will still get very hot, so be careful ! When it grips the cable firmly, take it out of the oven and allow to cool.

Next, bare the end of each strand of wire in the cable and separate them. Now, you must solder the cable to the D-connector as follows. If you look at the D-connector from behind you will see it has 9 receptacles for the wires and they are numbered as follows :-



Solder the coloured wires to the pins as follows (disregard the black wire as it is not used. You can also omit the white wire if you are only having two fire buttons as I suggest).

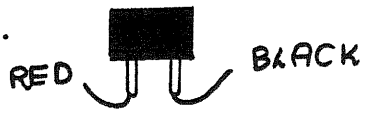
PIN	COLOUR	FUNCTION
1	Brown	Up
2	Orange	Down
3	Green	Left
4	Violet	Right
5	White	Spare (not needed)
6	Red	Fire 2
7	Yellow	Fire 1
8	Blue	Common 1
9	Grey	Common 2

Once you have done this you can assemble the cover making sure that it will still fit into the socket on the back of the computer.(WARNING !!! Never plug anything into the computer when it is switched on. We cannot be responsible for any damage to yourself or your computer through this project. It is working perfectly on our computer and no damage has been caused.) This is not as easy as it sounds and the only solution may be to use superglue and/or masking tape to secure the connector to the cover.

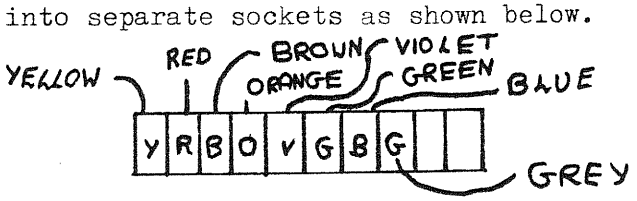
The next stage is to make up the box. You will need to drill a hole in the side of the box for the cable to enter through. On the lid of the box, mark out where the keys are going to go so that they are comfortable for you and then drill holes for the main body of the switch to fit into. Next put the switches in place and you can secure them with a blob of superglue so that they cannot turn or come out. You must also cut a slot in the top of the box for the slide switch to fit into and this can also be secured with superglue.

Wiring up

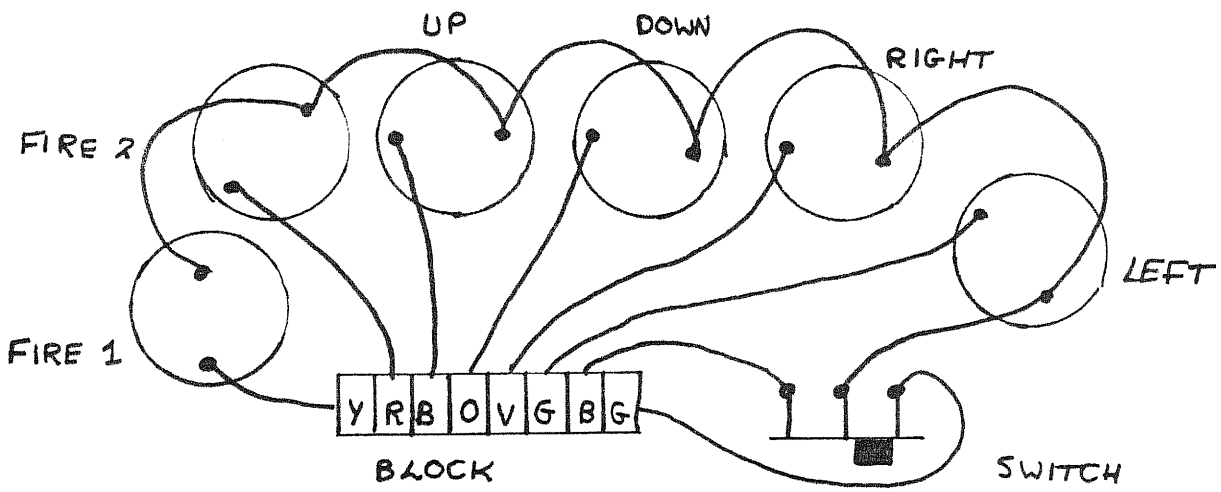
To start off with, solder two wires, one red and one black, each about 4 inches long onto the two legs of the keys as shown. NB It does not matter which wire is soldered to which leg.



On the toggle switch there are three legs. Solder a black wire to the centre leg and a red wire to each of the other legs. Take all of the black wires, including the one on the toggle switch and solder them together and wrap tape around the join.
Next take the terminal block and secure, by means of the screws, the wires from the ribbon cable into separate sockets as shown below.



Take the red wires from the switches and attach them to the other side of the block. Take the wire from Fire button 1 and attach it to socket number one and repeat with the other switches as shown below. Then do the same with the red wires from the toggle switch. When you have finished wiring up the Joyboard, screw the lid on, and secure the cable with tape or better still, with a cable clamp. For some finishing touches you can put some lettering on the box with transfers and also on the keytops.

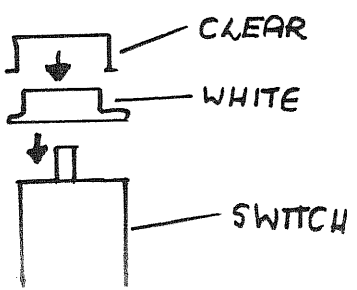


The keytops are assembled and fixed as shown below. The joyboard can be tested by using the short program below, which prints out two columns of numbers which should change when keys are pressed. If it works with the switch in one position, try it in the other and the other column should change. If both change at the same time, check the connections on the slide switch. The numbers are :- Left 4, Right 8, Up 1, Down 2 Fire 1 32, Fire 2, 16.

TEST PROGRAM

```
10 PRINT JOY(0),JOY(1)
20 GOTO 10
```

KEYTOP ASSEMBLY

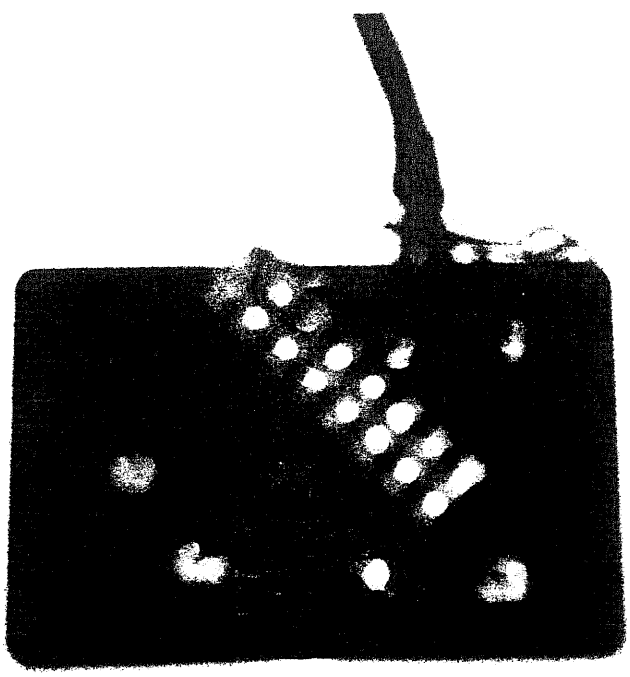


Parts —

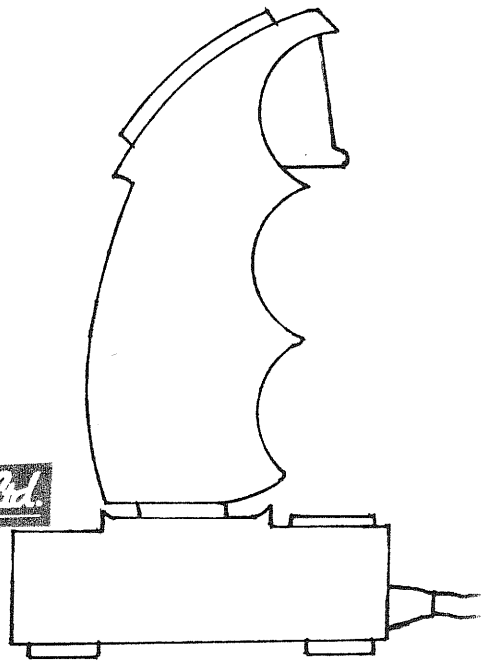
	NAME OF PART	ORDER CODE	QUANTITY	PRICE EACH	TOTAL
	D-Range 9 way socket	RK61R	1	48p	£0.48
*	D Hood 9 way	FP27E	1	45p	£0.45
	Ribbon Cable 10 way	XR06G	1	80p	£0.80
*	Heat Shrink CP 64	BF90X	1	£1.10	£1.10
	Keyboard Switch	FF61R	6	38p	£2.28
	Keytop 1 Position	FF62S	6	20p	£1.20
*	Stick-on Feet Square	FD75S	1	38p	£0.38
	Terminal Block 5A	HF01B	1	36p	£0.36
	ABS Box MB5	YN40T	1	£1.95	£1.95
	SP Slide	FF77J	1	20p	£0.20

TOTAL	=	£9.20
P + P	=	£0.50
GRAND TOTAL	=	£9.70

All these parts can be obtained from :- Maplin Electronics, PO Box 3, Rayleigh, Essex SS6 2BR, and cheques or postal orders should be made payable to 'Maplin Electronics Plc' and crossed.



GAMES REVIEWS



Emlyn Hughes International Soccer

by Audiogenic Software Ltd.

Up until recently the best football simulation has been, without a doubt, Matchday II from Ocean, but even this programming miracle has been surpassed by Emlyn Hughes International Soccer. This is a complicated and in depth game which allows you to manage a team, play against the computer or friend, or even do both. In the main menu section you are faced with many options which range from picking a team to changing the goalie's shirt colour. Each player has a different skill level for speed, defence and attack abilities and his fitness and this increases greatly the reality of the game. You can change the names of the players, their skill levels and substitute them as well as varying the length of the match from two minutes to a full ninety minutes and many other features.
When you actually get onto the pitch your players have an incredibly varied and large number of moves which they can perform including lobs, different strength kicks, headers, sliding tackles, penalties, barging and side-stepping as well as optional goalie control. The sound is fairly simple, just whistles and cheers, but the graphics are fast and incredibly detailed. It costs £9.95 (cass.) or £14.95 (disc).

TOM Although I am not a football fan myself, I found this game very addictive. The graphics were excellent during the game and information was displayed well. In the menu screen, control was easy using a system of pull-down menus. The game contained plenty of action, skill, planning and variation. It is a game which will keep you entertained for many hours. There are no problems with a green screen and although you need two joysticks for two player control this does not detract from the game as a whole. The sound was its only let down and the screen was a bit small. However, overall it is an excellent buy at £9.95 (cass.) or £14.95 (disc).

86/100

MARK It took me a while to get used to the format of the screen, having not played this type of game before. However, I soon realised what to do and was fairly impressed with the graphics which were quite detailed. The read out was clear and told you just the necessary facts. I found the game play confusing at first and it took me a while to master the controls but eventually I got the hang of it. The sound was not very good, just the odd whistle and cheer, but overall the game's not bad. If you are a football fan, you'll love it, if not it will grow on you. Excellent value for money.

82/100

JON The graphics in this game are good, it is addictive and the sound is alright. The only problem with the game is that it is very hard to control because of the number of different moves available (all of which are done very well !). It is sometimes difficult to tell who has the ball in some situations. It is not very original and is the best of all soccer games available. It is very addictive and I couldn't stop playing it. The penalties and fouls are excellent and there is a feeling of success as you score.

94/100

ADVENTURE GAMES

Time and Magik - a highly recommended compilation of three of Level 9's best. They are Lords of Time, Red Moon and the Price of Magik and it costs £14.95 (cass. & disc).
The Bard's Tale - the best ever role-playing game for the Amstrad. An excellent buy at £8.95 (cass.) or £14.95 (disc).

Released by : **CODEMASTERS**

Price : £2.99 tape

REVIEW - This is another game from Codemasters' budget range which has recently increased its price from £1.99 to £2.99. In it you are Blade Warrior, a fighter who must travel around Helfyre's Castle and its surroundings collecting various magical objects in the right order. The scenario is a typical one :-
Many years ago, the Evil Squire Helfyre ruled over the village of Loxton and caused great hardship and suffering (yawn!). The people of Loxton dragged him from his bed and killed him. Helfyre's manservant laid his remains in a secret tomb and he was reincarnated as the Death Demon. As the Death Demon he could carry out his evil plans. Two hundred years later you must reunite Helfyre's skull with the rest of his body before midnight. You must complete the mission in ten minutes and save the village of Loxton from an 'unholy orgy of violence and destruction'.

The graphics are good as is the sound (although there is no tune) and the game even boasts 'amazing sampled speech!' However, this is far from true as the only speech is a harsh, grating 'Ahh! Rest in peace' which is barely understandable. The directions are simple and easy to master and you are given the choice of a redefinable keyboard or joystick. It is basically a platform-and-ladders type game but is incredibly addictive. The sprites move smoothly and quickly. It is no innovation but is one of the best of its type.

MARK - The graphics are good and colourful but the scenery is not very original. It is very addictive and I found, as with other Codemasters games, excellent value for money at £2.99 but I do wish that they could think of an original storyline. The only point that detracted from the game was the sound as there was no tune and the sampled speech was terrible. However, it is a bargain! 73%

JON - The graphics are edging on incredible and its only let down is the sound. It is very addictive and when you die (as I did frequently!) it is really frustrating. Anyone with curiosity and persistance will enjoy it. I did! 81%

TOM - Very rarely have I played a game which is as addictive as this. It is incredible. What sound there is, is done well but there isn't enough of it. The graphics are smooth and colourful. Recommended. 76%

JOYSTICK SURVEY

As I am sure you know, joysticks are an important, though not essential, ingredient in any computer set up. They are mostly used in conjunction with games software, but can be used as a form of 'mouse' - an alternative to the keyboard. Thus a joystick, in its simplest terms, is really a form of input device. The three I am going to look at and review are :-

1) The JY2 - the Amstrad Joystick
2) The Arcade - made by a Dutch firm
3) The Cheetah 125+

JY2 —— This joystick comes free with any of the Amstrad computer packages although if bought individually it costs £14.95. It looks a compact joystick, made black plastic with two red fire buttons on the handle. However, it looks of poor quality and the handle is far too small and uncomfortable to hold. The fire buttons were reasonable and responded with a definite click, but were very small. The handle seemed rather loose and I thought that one tug in any direction would force it out of its socket, in fact I was just waiting for the whole thing to fall apart. The one point that I did like was the suckers on the base of the unit for grip. The JY2 also includes a facility for using two joysticks simultaneously. However very few games or utilities actually allow the user to utilise this feature. So to sum up :-

FOR - Suckers on the bottom, two fire buttons.

AGAINST - Bad quality and bad response, high price - £14.95

RATING - 3 out of 10, not recommended.

3/10

"The Arcade" ——— This joystick also looked compact but immediately it struck me that it was of better quality. The handle was of a different design to the previous one. It had a 'ball' at the top of the stick for the hand to grip round and, surprisingly, this worked well and it felt very comfortable. The joystick had only one large fire button, but this was situated in the centre of the base of the joystick for left or right handed persons. The thing I liked especially was the loud audible click the handle made when it was pushed in any of the eight directions. This relieved the temptation to 'yank' on the handle to make sure it had made contact. The only problem with this is that after prolonged periods of time it can become very annoying and distracting. The only fault of the joystick is the grip on the surface. It did not have suckers like the JY2, but had three small rubber discs that did not secure it firmly to any surface. This joystick actually looked stylish in its black and red colours and was very pleasing to use. It did not have the additional expansion port like the JY2 but this does not detract greatly from the joystick. However, as it is Dutch made it is quite hard to find and its price fairly high - £14.95. To sum up :-

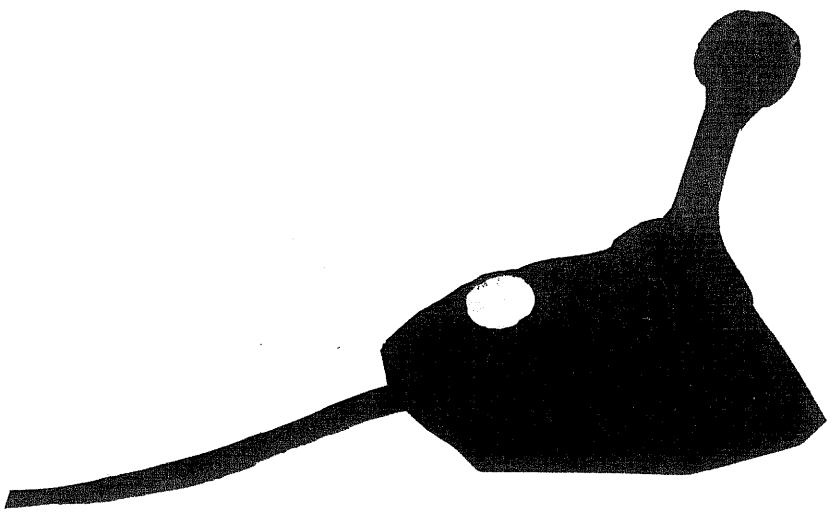
FOR - Style and quality.

AGAINST - Bad grip and high price - £14.95.

RATING - 8 out of 10, recommended but a bit expensive.

8/10

If any reader knows a shop which has a regular supply of these joysticks, we would be very grateful if they could tell us, so we can inform our other readers.



Before buying a joystick it is well worth trying it out yourself and looking at all the different types of joystick available. The handle must be one which fits your hand well and is comfortable. It is very important that the fire buttons are in suitable places for you and not in places which are impossible to use.

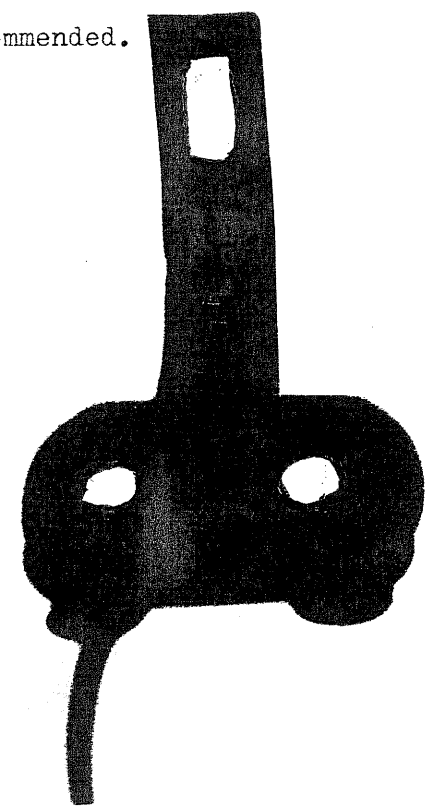
Cheetah 125+ —— The third joystick is of a very different design to the others, being larger and having more features. These include having four, large red fire buttons; two on the handle and two on the base. This obviously gives more choice to the user, whether he (or she) is left or right handed. This means that it is easy to obtain the most comfortable button for any individual. There are large grips on the handle into which the hand can fit easily. As with the JY2 there are suckers on the base to keep it stable on any surface. The only feature that isn't available is the click action but the many other features more than make up for this loss. The Cheetah 125+ also features an autofire selecter but this works with only partial success on the Amstrad. It is a fact that only the Amstrad is adverse to the use of an autofire function and no other computer. The joystick has to connecters available and this gives a wider range of computers that it will work on. It is robust, sturdy and fairly cheap and is an excellent buy. So to sum up :-

FOR - Quality, four fire buttons, suckers and price - £8.95

AGAINST - No click action.

RATING - 9 out of 10, highly recommended.

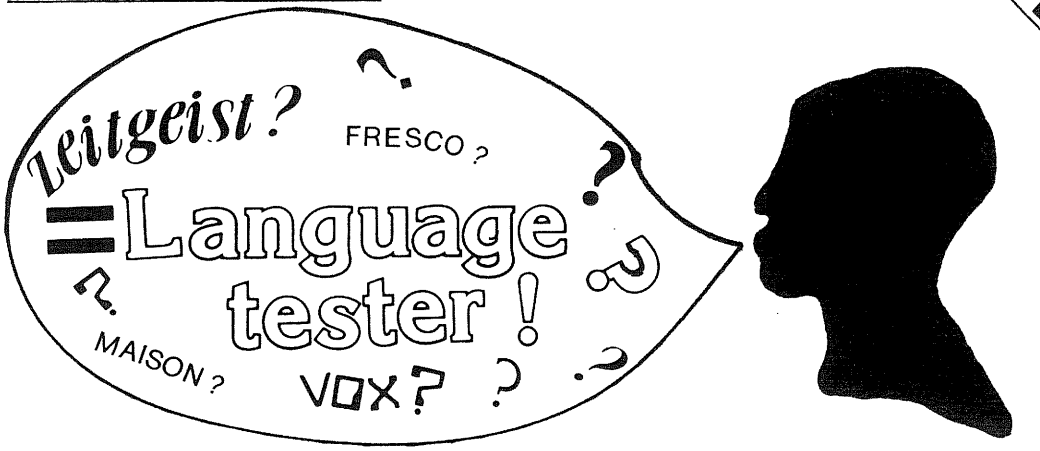
9/10



Suppliers ——— JY2 - Amsoft Mail Order, Enterprise House, PO Box 10, Roper Street, Pallion Industrial Estate, Sunderland SR4 6SN. Tel. (091) 5108787.

Cheetah 125+ - Cheetah Marketing Ltd., Norbury House, Norbury Road, Fairwater, Cardiff CF5 3AS. Tel. (0222) 555525.

PROGRAM 2



If you have ever had difficulty learning foreign languages, words or phrases then the 'Language Tester' program is for you. We have tried to make it as full and complete as possible but if we had included too many features it would just have been too long to publish. However, because the program is divided up into numerous subroutines it should not be too hard to add your own personal touches; we look forward to seeing your additions to it.

As it is you are faced with five main choices, and they are :-
LOADING a file, CREATING a file, CHOOSING a language, TESTING yourself and also QUITTING the program.

LOAD a file - This allows you to load in one of your previously prepared files of vocabulary and phrases.

CREATE a file - This enables you to create your own file of vocanulary and to save it to tape (or disc) for future use.

CHOOSE a language - Using this you can select your language (French and German are built in, complete with extra letters) and if necessary, to load in any additional letters required.

TEST on words - The computer can test you on any of the words in the loaded file, either from or to English.

QUIT program - Allows you to exit the program.

By pressing ESC twice (except whilst it is waiting for a typed input) you are returned to the main menu of the program. During loading, saving and creating files follow the on screen instructions and press any key to continue the program.

When loading in your own letters make sure that it is renumbered to 2090 and that the first line reads - SYMBOL AFTER 2Ø.

OOPS— Even we are not infallible and several errors have crept into the program. They are :-

```
86Ø a$=INKEY$:IF a$="" THEN 86Ø
187Ø SYMBOL 96,24,1Ø2,56,24,24,24,6Ø,Ø
```

```
 10 REM Language Tester (c) 1989, T.Defoe
 20 GOSUB 60:REM Initialisation
 30 GOSUB 120:REM Screen layout
 40 GOTO 250:REM Main screen menu
 50 END
 60 REM Initialisation
 70 DIM word$(1000,2)
 80 lang$="":in$="":a$=""
 90 file$="":where=0
100 ON BREAK GOSUB 1440
110 RETURN
120 REM Screen layout
130 MODE 2:CALL &BC02
140 WINDOW #2,4,76,2,6:PAPER #2,1:PEN #2,0:cls #2
150 WINDOW #1,4,76,18,23:PAPER #1,0:PEN #1,1:CLS #1
160 WINDOW #0,4,76,8,16:PAPER #0,0:PEN #0,1:CLS #0
170 BORDER 7:MOVE 0,0:DRAW 0,399,1:DRAW 639,399
180 DRAW 639,0:DRAW 0,0
190 LOCATE #2,28,2:PRINT #2,"Language Tester"
200 LOCATE #2,26,3:PRINT #2,"(c) 1989, T.Defoe"
210 LOCATE 21,2:PRINT "Welome to the LANGUAGE TESTER"
220 LOCATE 20,3:PRINT "Please PRESS any key to continue"
230 GOSUB 1410
240 RETURN
250 REM Main Screen Menu
260 in$=""
270 CLS #0:CLS #1
280 x=LEN(lang$):y=(72-x)/2:LOCATE #2,y,4
290 PRINT #2,UPPER$(lang$)
300 where=0:in$=""
310 LOCATE 27,2:PRINT "1) Load new file"
320 LOCATE 27,3:PRINT "2) CREATE new file"
330 LOCATE 27,4:PRINT "3) SELECT language"
340 LOCATE 27,5:PRINT "4) TEST on words"
350 LOCATE 27,6:PRINT "5) END program"
360 LOCATE 22,8:PRINT "What is your choice (1-5)?"
370 know$="12345"
380 in$=INKEY$:IF in$=""THEN 380
390 where=INSTR(know$,in$)
400 IF where=0 THEN 360
410 ON where GOSUB 430,640,890,1100,1400
420 GOTO 250
430 REM Load new file
440 CLS #1:CLS #0
450 IF lang$="" THEN LOCATE 25,3:PRINT "CHOOSE LANGUAGE FIRST":GOSUB 1410:GOTO
    250
460 LOCATE 21,3:PRINT "What is your choice of file?"
470 LOCATE #1,6,2:INPUT #1,"Enter file NAME :- ",file$
480 IF LEN(file$)>5 THEN CLS #1:GOTO 470
```

```
490 LOCATE #1,6,3:PRINT #1,"Press PLAY and any key"
500 GOSUB 1410
510 file$="!"+file$
520 OPENIN file$
530 INPUT #9,num
540 FOR i=1 TO num
550 INPUT #9, word$(i,1)
560 INPUT #9,word$(i,2)
570 NEXT i
580 CLOSEIN
590 CLS #1:CLS #0
600 LOCATE 26,3:PRINT "FILE IS NOW LOADED!"
610 LOCATE 23,4:PRINT "Press any key to continue"
620 GOSUB 1410
630 RETURN
640 REM Create new file
650 CLS #0:CLS #1
660 IF lang$="" THEN GOSUB 1460
670 IF file$<>""THEn GOSUB 1510
680 CLS #0:CLS #1
690 LOCATE 17,3:PRINT "How many WORDS do you want to enter?"
700 LOCATE #1,6,3:INPUT #1,"Number of words:- ",num
710 IF num>1000 THEN GOTO 680
720 IF num<1 THEN GOTO 250
730 i=1
740 IF i-1=num THEN GOSUB 1590:RETURN
750 CLS #0:CLS #1
760 LOCATE 30,8:PRINT "WORD:- ";i
770 LOCATE 4,2:PRINT "Enter ENGLISH word followed by the  ";
780 PRINT  lang$;"  word."
790 LOCATE #1,6,3:INPUT #1,"Enter ENGLISH word :- ",word$(i,1)
800 word$(i,1)=LOWER$(word$(i,1))
810 CLS #1:LOCATE  #1,6,3:PRINT #1,"Enter  ";lang$;
820 INPUT #1,"  word  :- ";word$(i,2)
830 word$(i,2)=LOWER$(word$(i,2))
840 LOCATE 4,4:PRINT "Are  ";word$(i,1);"  and  ";
850 PRINT word$(i,2);"  both correct  (y/n) ?"
860 a$=INKEY$:a$="" THEN 860
870 IF UPPER$(a$)="Y" THEN i=i+1: GOTO 740 ELSE GOTO 740
880 RETURN
900 REM Choose language
910 CLS #0:CLS #1
920 LOCATE 26,2:PRINT "Choose the LANGUAGE"
930 LOCATE 31,4:PRINT "1) FRENCH"
940 LOCATE 31,5:PRINT "2) GERMAN"
950 LOCATE 31,6:PRINT "3) OTHER"
960 know$="123"
970 in$=INKEY$:IF in$="" THEN 960
980 where=INSTR(know$,in$)
990 IF where=0 THEN 960
```

```
 990 ON where GOSUB 1770,1900,1010
1000 RETURN
1010 REM Other Language
1020 CLS #0:CLS #1
1030 LOCATE 25,3:PRINT "What is the language?"
1040 LOCATE #1,6,2:INPUT #1,"Name of LANGUAGE :- ";lang$
1050 lang$=UPPER$(lang$)
1060 LOCATE 5,4:PRINT "Does ";lang$"  require any special letters (y/n)?"
1070 a$=INKEY$:IF a$="" THEN 1070
1080 IF UPPER$(a$)="Y" THEN GOSUB 1990
1090 RETURN
1100 REM Language Test
1110 CLS #0:CLS #1
1120 IF word$(1,1)="" THEN LOCATE 28,3:PRINT "Load file first":GOSUB 1410:GOTO
      250
1130 LOCATE 22,3:PRINT "Do you want it to be from:-"
1140 LOCATE 27,5:PRINT "1) ENGLISH to  ";lang$
1150 LOCATE 27,6:PRINT "2) "lang$;"  to ENGLISH"
1160 know$="12"
1170 in$=INKEY$:IF in$="" THEN 1170
1180 where=INSTR(know$,in$)
1190 IF where=0 THEN GOTO 1170
1200 ON where GOSUB 1220,1310
1210 RETURN
1220 REM English to other
1230 score=0
1240 FOR i=1 TO num
1250 CLS #0:CLS #1
1260 LOCATE 6,3:PRINT "What is  ";word$(i,1);"  in  ";lang$;" ?"
1270 LOCATE #1,6,3:INPUT #1,"It is :- ";an$
1280 IF LOWER$(an$)=word$(i,2) THEN GOSUB 2290 ELSE GOSUB 2370
1290 NEXT i
1300 RETURN
1310 REM Other to English
1320 score=0
1330 FOR i=1 TO num
1340 CLS #0:CLS #1
1350 LOCATE 6,3:PRINT "What is  ";word$(i,2);"  in ENGLISH  ?"
1360 LOCATE #1,6,3:INPUT #1,"It is :- ";an$
1370 IF LOWER$(an$)=word$(i,1) THEN GOSUB 2290 ELSE GOSUB 2370
1380 NEXT i
1390 RETURN
1400 MODE 2:END
1410 REM Key Test
1420 a$=INKEY$:IF a$="" THEN 1410
1430 RETURN
1440 REM Problem?
1450 GOTO 250
1460 REM Cannot do!
1470 CLS #0:CLS #1
```

```
1480 LOCATE 21,3:PRINT "Please CHOOSE a language first"
1490 LOCATE 23,4:PRINT "Press any KEY to continue"
1500 GOSUB 1410:GOTO 250
1510 REM Warning!!
1520 CLS #0:CLS #1
1530 LOCATE 19,3:PRINT "WARNING!! Other FILE will be lost"
1540 LOCATE 14,4:PRINT "Press ESC twice to return to the MAIN MENU or"
1560 LOCATE 23,5:PRINT "Press any KEY to continue"
1570 GOSUB 1410
1580 RETURN
1590 REM save file
1600 CLS #0:CLS #1
1610 LOCATE 18,3:PRINT "What do you want to CALL the file?"
1620 LOCATE #1,6,3:INPUT #1,"Enter FILENAME 1:- ";file$
1630 IF LEN(file$)>5 THEN GOTO 1600
1640 file$="!"+file$
1650 LOCATE #1,6,4:PRINT #1,"Press REC and PLAY, then any key"
1660 GOSUB 1410
1670 OPENOUT file$
1680 PRINT #9,num
1690 FOR i=1 TO num
1700 PRINT #9,word$(i,1)
1710 PRINT #9,word$(i,2)
1720 NEXT i
1730 CLOSEOUT
1740 CLS #0:CLS #1
1750 LOCATE 23,4:PRINT "Press any KEY to continue"
1760 GOSUB 1410:RETURN
1770 REM French letters
1780 lang$="FRENCH"
1790 SYMBOL AFTER 20
1800 SYMBOL 64,96,24,120,12,124,204,118,0
1810 SYMBOL 124,24,96,120,12,124,204,118,0
1820 SYMBOL 91,24,96,60,102,126,96,60,0
1830 SYMBOL 123,96,24,60,102,126,96,60,0
1840 SYMBOL 93,24,102,60,102,102,102,60
1850 SYMBOL 125,24,102,60,102,126,96,60,0
1860 SYMBOL 92,24,102,120,12,124,204,118,0
1880 SYMBOL 94,0,60,102,96,102,60,4,28
1890 RETURN
1900 REM German Letters
1910 lang$="GERMAN"
1920 SYMBOL AFTER 20
1930 SYMBOL 94,120,198,198,252,198,198,248,192
1940 SYMBOL 91,36,0,60,102,126,96,60,0
1950 SYMBOL 93,36,0,120,12,124,204,118,0
1960 SYMBOL 123,36,0,60,102,102,102,60,0
1970 SYMBOL 125,36,0,102,102,102,102,62
1980 RETURN
1990 REM Other letters
```

```
2000 CLS #1:CLS #0
2010 LOCATE 14,3:PRINT "Make sure the letters are RENUMBERED to 2090"
2020 LOCATE 25,4:PRINT "What is its filename?"
2030 LOCATE #1,6,3:INPUT #1,"Enter filename  :- ";name$
2040 IF LEN(name$)>5 THEN GOTO 2010
2050 name$="!"+name$
2060 LOCATE 25,5:PRINT "Press PLAY and any key"
2070 GOSUB 1410
2080 CHAIN MERGE name$,2090
2090 REM
2280 GOTO 250
2290 REM Correct
2300 CLS #1:CLS #0
2310 LOCATE 31,3:PRINT "CORRECT!!"
2320 score=score+1
2330 LOCATE 30,4:PRINT "SCORE :- ";score"
2340 LOCATE 23,5:"Press any key to continue"
2350 GOSUB 1410
2360 RETURN
2370 REM Wrong
2380 CLS #1:CLS #0
2390 LOCATE 32,3:PRINT "WRONG!!"
2400 LOCATE 6,4:PRINT word$(i,2);"  is  ";word$(i,1);"  in English"
2410 LOCATE 30,5:PRINT "SCORE:-  ";score
2420 LOCATE 23,6:PRINT "Press any KEY to continue"
2430 GOSUB 1410
2440 RETURN
```

Below are printed the two sets of extra foreign letters which are built into the program and also which keys are pressed to access them.

FRENCH

[	=	é
SHIFT [	=	è
]	=	ô
SHIFT]	=	ê
\	=	à
SHIFT \	=	î
^	=	ç
@	=	ù
SHIFT @	=	á

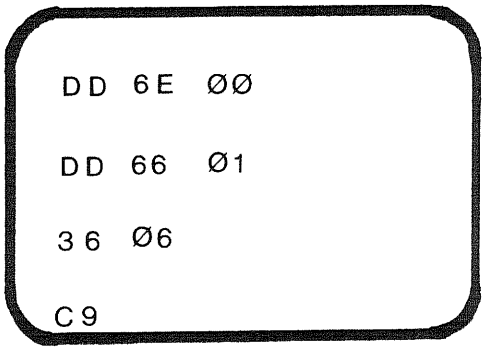
GERMAN

[	=	ö
SHIFT [	=	ü
]	=	ä
SHIFT]	=	ß
^	=	ß

It is suggested that, when you are designing your own letter sets, you use only the keys listed above as they are not used in the program in any way at all. You can find their ASCII codes in the appendices of your manual.

MACHINE CODE

Part 1



Far too many magazine series and books about machine code dive straight into pages and pages of numbers and mnemonics without even explaining what machine code is and what it does. In this series we will hopefully answer these questions and take you on a step by step guided tour of the mysterious land of 'Machine Code'. We will try to include many example programs which will illustrate the point in hand rather than giving you just a jumbled list of names and numbers.

What is machine code ? With BASIC the computer has to translate all of your commands into numbers which it can understand and execute and this slows it down considerably. By using machine code you have to enter just numbers which the computer can understand without having to translate them. Thus it runs quicker and you have more functions available than in BASIC. The disadvantage is that you need to know what the numbers mean. However, by using an assembler you can overcome this problem.

There are two ways of getting machine code programs to run on the Amstrad. The more expensive, but better, method is to buy an assembler and the cheaper way requires no additional purchases. However, if you wish to pursue machine code further than just typing in listings from magazines it would be very worth while if you did acquire an assembler.

What is an Assembler ? An assembler allows you to enter mnemonics instead of pure numbers and changes the mnemonics into numbers for you. One of the best on the market is MAXAM from Arnor. It costs £19.95 on tape, £26.95 on disc and £39.95 on ROM chip and you can contact Arnor at Protext House, Wainman Road, Peterborough PE2 OBU. MAXAM also includes a memory editor, a disassembler and a host of other features and is the one that we use for compiling the listings used here. However for people without an assembler we will also print a BASIC 'poker'.

How do I run the programs ? To run the BASIC poker all you have to do is type it in and run it. Whenever you want it to work all you do is type CALL &4ØØØ. Similarly, when using MAXAM, get into the Text Editor mode and type in the mnemonics, press ESC and tell it to Assemble Text (A). Then press Q followed on the next menu by B to get into BASIC and whenever you want it to work just type CALL &4ØØØ.

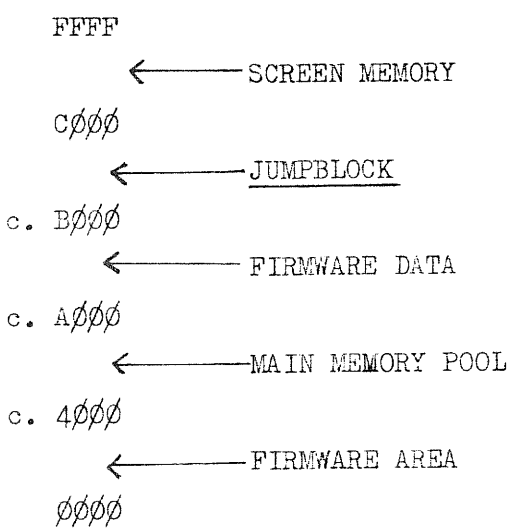
What do I need to know before I start ? All you need to know is how and why Binary and Hexadecimal work and how to convert them. If you are unsure on this it might be an idea to read the section 'What Is My Amstrad?' for a full explanation.

Registers - In BASIC, variables are very common and you would find programming all but impossible without them. Likewise in machine code, you have variables which are known as REGISTERS. You have several special registers and six general ones and this may sound a bit limiting but there are ways round it.

The six general registers are B, C, D, E, H and L. Each of them can hold a number between 0 and 255 and are thus 8-bit registers. However, they can be grouped together in register pairs (BC, DE, HL) which can represent numbers from 0 to 65535 and are 16 bit numbers. The main use of all general registers is for data storage but certain registers are used for certain things.

There are quite a few special registers but the two we will look at are A and F. The register A is the ACCUMULATOR and is used for arithmetic and other logical operations. F is the FLAG REGISTER and is used to hold information about the results of other operations. Both of these are used only as 8-bit registers, although for some other operations it is useful to group them together as a register pair, AF.

THE CPC MEMORY MAP :- The CPC's memory is split up as follows.



The memory of your Amstrad is not very easy to understand because of a piece of hardware called the GATE ARRAY. This allows the 32K of ROM to overlap the 64K of RAM. The first 16K and last 16K of RAM are shared by the ROM and this is why the memory map, when shown in more detail, may look confusing. The screen memory is self explanatory as is the main memory pool. The two sections of the memory map which include the word firmware are where most of the computer's thinking and logic takes place. However, the most interesting part is the JUMPBLOCK. There are two parts to the jumpblock. One bit goes from B9ØØ to about B921 and the longer chunk starts at about BBØØ and goes on until about BEØØ. These addresses are called whenever we want the computer to do something. What they do is send the computer to a routine deep inside the firmware.

NOTE : All of the addresses above are given in hexadecimal.

There are two main reasons for the jumpblock, and they are :-

i) So that they are all in a block and so the codes are easier to remember.

ii) As long as the jumpblock remains in the same place, most programs written for the CPC 464 will also work on the 664 and 6128.

However, before you can use these routines you need to load certain registers with numbers. The best way to find out which routines are available, what address they are at, what entry conditions there are and which variables they corrupt is to study the Firmware Guide from Amsoft at £19.95. It had no ISBN number and is now out of print. If you can get your hands on a copy, buy it (and get me one!), because it is invaluable. The next best thing is the 'Advanced Users-Guide' by Daniel Martin from Glentop (ISBN 1-85181-122-2, price £8.50). Although it does not go into so much of the details, it still contains the main firmware calls, a brief description of each and the entry and exit conditions.

Your **FIRST** machine code program

The most important mnemonic in Assembly language is RET and its hexadecimal value is C9. In future new commands will be shown as follows:-

RET c9

From this you can see one important thing already. That is,that hexadecimal numbers can be entered in either upper or lower cases. So what? you may be asking. What does it do? Well, RET is an abbreviation of RETurn and does just that. It has two uses. The first tells the computer to return from a subroutine (similar to BASIC). Its second use is to tell the computer to return from machine code back into BASIC (or whatever language it was called from). For people with assemblers it is very simple:-

Type in - ORG &4ØØØ

RET

Get into BASIC and type, CALL &4ØØØ. With any luck nothing will happen and the cursor will appear as normal. For people without an assembler it is slightly more tricky :-

Type in the program below :-

```
10 REM Basic Poker (c) 1989, T.J.Defoe
20 REM Type in hexadecimal codes as shown
30 REM (Note that spaces should be disregarded
40 REM and numbers can be typed in seperately)
50 REM Type 'END' to finish data entry and in
60 REM BASIC type 'CALL &4000' to make it work.
70 REM
80 REM
90 addr=&4000
100 INPUT "What is the hexadecimal number";a$
110 IF UPPER$ (a$)= "END" THEN END
120 x=LEN(a$)/2
130 y=INT(LEN(a$)/2)
140 IF x<>y THEN PRINT "There is an error": GOTO 100
150 b$=LEFT$(a$,2)
160 z=VAL("&"+b$)
170 POKE addr,z
180 addr=addr+1
190 a$=RIGHT$(a$,LEN(a$)-2)
200 IF a$="" THEN GOTO 100 ELSE GOTO 150
```

When you RUN it, it will ask you to enter a hexadecimal number followed by the ENTER key. In the example program you should enter C9, although in longer programs there may be more than one pair to enter (in this case omit any spaces). When you have finished entering the numbers, you should type END instead to exit the program. To make the routine work you should type :- CALL &4000 and it will work as described.

PRINT - A long time ago, when you were just starting to learn BASIC, you probably came across a very simple, useful and not very interesting command, PRINT. However, in machine code it is still useful and not very interesting, but it is NOT simple. Therefore to start off with, we are only going to look at printing single letters.

The firmware call for printing a letter to the screen is at &BB5A in the jumpblock. To print a letter you have to first load the accumulator with the ASCII (American Standard Code for Information Interchange) number of the character you wish to print. Luckily, all of the ASCII codes are printed at the back of your computer's manual and so they can be found out easily. Thus the program below prints the letter A when cal d.

```
ORG &4ØØØ
LD  a,65          3E 41
CALL &BB5A        CD 5A BB
RET               C9
```

In this there are two new mnemonics and one directive and they are explained below :-

ORG - This tells the computer at what address in its memory that the machine code program should be stored. With the BASIC poker it is not required because the program is told where to poke the data at the beginning.

LD - This is equivalent to the BASIC 'LET' command and gives a register a certain valu

CALL - We have already met CALL, but not as a mnemonic. As before, its purpose is to CALL a routine in memory. In this case it is the hardware printing routine.

Below is an explanation of the program which also points out a number of other poi

```
ORG &4ØØØ      ;  This tells the routine where to be located in the
                   computers memory (eg. &4ØØØ).
LD  a,65        ;  This lets A equal 65. However, you will notice that
                   the 'poker' stores it as 3E 41. 3E is the hexadecimal
                   code for LD a, and 41 is hexadecimal for 65.
CALL &BB5A      ;  This tells the computer to go to the hardware printi
                   routine and print whatever character is in A. The
                   'poker' stores it as CD 5A BB. The CD represents CAL
                   and the number BB5A is stored as two hexadecimal numb
                   ers with the low byte first.
RET             ;  This tells the program to RETurn to BASIC.
```

The computer, whether or not you have an assembler, stores the machine code program

in its memory as the hexadecimal numbers that are shown above. All the assembler does is translate your easy-to-understand mnemonics into hexadecimal numbers.

The routine below prints the word 'Hello' onto the screen and includes several new commands which are explained afterwards.

```
                ORG &4000
                .txt_output equ &BB5A
                ld hl,string            21 12 40
                call print              CD 07 40
                ret                     C9

                .print
                ld a,(hl)               7E
                cp 0                    FE 00
                ret z                   C8
                call txt_output         CD 5A BB
                inc hl                  23
                jp print                C3 07 40

                .string
                db "Hello"              48 65 6C 6C 6F
```

CP - This ComPares A with 0 (or any other number specified). If they are the same then Z (zero) flag is set to 0 otherwise it is set to 1.

RET Z - This returns if the Z flag is 0 otherwise it allows the program to carry on.

INC HL - This INCrements HL by 1 and is equivalent to HL=HL+1.

JP - This tells the computer to JumP to either a memory location or a label (which denotes a memory location). In this case it tells the computer to jump to the label 'print' which is at the beginning of the routine.

DB - Allows you to store data in the form of letters (with MAXAM you can also use TEXT in exactly the same way).

.print, .string and .txt_output are all labels and are just an easier way of pointing to an address in the computer's memory.

```
        LD HL,string        ;   loads HL with the address of the label .string
        LD A,(HL)           ;   loads A with the contents of the address stored
                                in HL. In this case it is the word 'Hello'.
.txt_output equ &BB5A       ;   it is usual machine code practice to put all
                                the addresses used, at the beginning of the
                                program and then to refer to them by their label
                                    eg.     call txt_output.
```

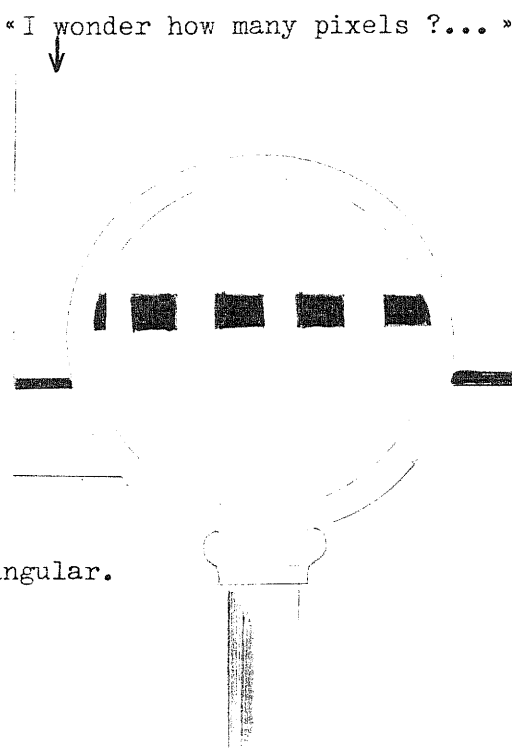
ADVANCED BASIC

GRAPHICS

INTRODUCTION - It is important at the beginning to understand what makes up the picture you see on your monitor without going into the details of the hardware. The picture that you see is made up of hundreds of tiny dots called PIXELS which can be set to the required colour. The Amstrad CPC has a grid of pixels 640 x 200 in size. The state of the pixels is continuously being updated and stored in the computer's memory. In MODE Ø you can display sixteen of the 27 colours available simultaneously but you have much 'chunkier' letters. This is because in MODE Ø, although you still have 640 x 200 pixels, the pixels are lit horizontally in groups of four. So the actual number of pixels you can separately address is only 160 x 200. Likewise in MODE 1, you can display 4 of the 27 colours at the same time but can control 320 x 200 pixels separately because they are lit in horizontal groups of two. Whereas in MODE 2 you can only display two colours simultaneously but you can control the full 640 x 200 pixels. For this reason Mode 2 is called the HI-RES mode (High Resolution Mode) and Mode Ø is called the LO-RES (Low Resolution) mode. If you have a powerful magnifying glass you can prove this to yourself, all you have to do is enter :-

```
MODE 0 : PLOT 1,1
```

- and count the number of pixels horizontally there are and then repeat with MODEs one and two.



The reason for the changing number of pixels is so that more and more colours can be used in that MODE. This works because spare memory is produced which can store more information about the colour of the pixel rather than saying that it is simply either on or off. However to stop you becoming at all confused by the changing of the screen pixels your Amstrad uses the grid size 640 x 400 in all MODEs and does the necessary calculations for you. But you may be saying, where does that 400 come into it? The answer is simple, it is like that to keep square dots looking vaguely square rather than rectangular.

GRAPHICS COMMANDS

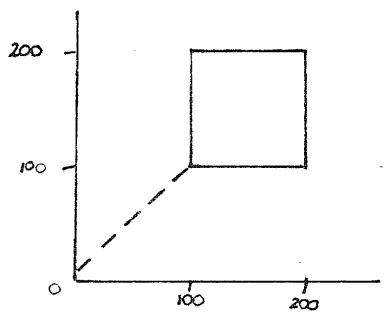
Almost all graphics commands have at least two numbers following them. These are the x and y coordinates of points on the screen and can have values as follow :-

x = 0 - 640 ; This is the horizontal coordinate
y = 0 - 400 ; This is the vertical coordinate.

MOVE - This allows you to move the graphics cursor about the screen without leaving a line on the screen. When you switch on the computer, or execute a MODE command the graphics cursor starts at point 0,0. The MOVE command is followed by the x and y co-ordinates of the point which it is to move to. eg. MOVE 200,100

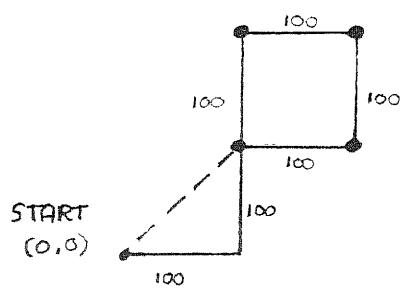
DRAW - This allows you to move the graphics cursor about the screen and draws a line from its previous location to its new one, which is shown by the x and y coordinates which follow it. eg. DRAW 300,500. The DRAW command has another optional parameter which tells it which pen to use for drawing the line. eg. DRAW 100,100,2. On a 664 or 6128 you have an optional 'INK MODE' parameter which we will discuss later (as well as how to emulate it on a 464). With the DRAW and MOVE commands you are moving or drawing to a place on an imaginary grid. So to draw a box :-

```
MOVE 100,100
DRAW 100,200
DRAW 200,200
DRAW 200,100
DRAW 100,100
```



RELATIVE COMMANDS - However, you can move and draw RELATIVE to the last point using the commands DRAWR and MOVER. So to use these commands to draw the same box :-

```
MOVER 100,100
DRAWR 100,0
DRAWR 0,100
DRAWR -100,0
DRAWR 0,-100
```



The first instruction tells the computer to move 100 up and 100 right from its start position of 0,0. Then it is told to move and draw 100 right and 0 up. It then draws to a point which is 0 right and 100 up. The minus sign in front of the next number means that the computer should draw from right to left, or from up to down (if it was the second number) instead of the other way round. So the next command means draw left 100 and up 0 and the last instruction says draw 0 right and 100 down. DRAWR also has the same additional parameters as DRAW. Before we go any further it would be very useful if you clearly understood these commands.

YPOS and XPOS - If at any time you wish to find out where the graphics cursor is, all you have to do is type :- PRINT XPOS,YPOS and the x coordinate of the graphic cursor followed by the y coordinate will be returned.

PLOT - This is similar in form to DRAW. However, it only lights the last (destination) pixel. eg. PLOT 320,200. As with DRAW it also has an optional ink colour parameter. eg. PLOT 163,356,3. (If the ink colour is omitted, it defaults to the last used colour As with DRAW it can also have an optional 'INK MODE' statement. An example program is given below. It draws a circle with a radius, r, and a centre, x and y. It calculates the position of the circle's edge in line 70 using SIN and TAN. Once you have seen it work, try omitting line 50 and see if you can spot the difference.

```
10 REM r=radius, x and y are the centre of circle
20 x=320:y=200:r=50
30 MODE 1
40 FOR i=1 TO 360
50 DEG
60 MOVE x,y
70 PLOT x+r*COS(i),y+r*SIN(i)
80 NEXT i
```

PLOTR - This is similar to PLOT except that it works on relative pixels. For a more detailed explanation on RELATIVE commands see MOVER and DRAWR. Like PLOT only the last pixel is lit and it can also have INK MODEs and colour statements. eg PLOTR -3,6. As a challenge, see if you can convert the above program to work using PLOTR !!

TEST - This command is useful for telling you the value of the INK at a certain point on the screen. However, its formation is slightly different from other graphic commands as the coordinates are contained in brackets. eg. PRINT TEST(300,100). Thus it is often used as a simple method of collision detection in BASIC games, as shown here

```
10 FOR i=1 TO 640 STEP 2
20 PLOT i,200
30 x=TEST(i+1,200)
40 IF x<>0 THEN END
50 NEXT i
```

To make it work type, MODE 1: MOVE x,y: DRAW x,y: RUN, using your own values instead o x and y. If the line you drew intercepts the line that the computer is drawing, the computer's line will stop where it meets, otherwise it will carry on until it reaches the far side of the screen. At present it only works in MODE 1, to make it work in MODE 0 change line 10 to - FOR i=1 TO 640 STEP 4
and to work in MODE 2 - FOR i=1 TO 640
The reason for this has already been explained. Another short example is shown below. When you run it, it will mirror the left side of a MODE 0 screen on the right side.

TESTR - This is the relative form of TEST and does the same thing except that it works on relative coordinates. eg. TESTR(0,-150). As with TEST it returns the value of the PEN that is found at the location under test.

ORIGIN - This command allows you to move the start location of the graphics cursor to anywhere on the screen and thus the coordinates are changed accordingly. The ORIGIN command is reset by the MODE command. This means that if you set the ORIGIN as being 100,35 and then execute a MODE command the ORIGIN will be reset to 0,0.
eg. ORIGIN 320,200 changes the points from (a) to the coordinates (b).

ORIGIN will be explained further next month and we will introduce INK, PEN, PAPER, BORDER, WINDOW, CLS and CLG. But now we turn to that INK MODE parameter.

INK MODE PARAMETER - On a 664 or 6128, DRAW, DRAWR, PLOT, PLOTR, MOVE and MOVER all have an additional ink mode parameter which can range from 0 to 3. However, on the 464 this function is not available but luckily we can emulate it very simply. The line below should only be included once at the beginning of a routine or program.

PRINT CHR$(23)+CHR$(n) ; where 'n' is the Ink Mode required.

The Ink Modes are :-

Ø	Normal
1	XOR
2	AND
3	OR

The most interesting of them is XOR. What it does is to make, when it meets another pixel, any lit pixels UNLIT and any unlit pixels LIT. Thus it performs an eXclusive OR on every pixel in its path. XOR's main use is to prevent something being erased when a picture passes in front of the background. This simple program illustrates XOR well, by drawing lines and then erasing them again very quickly.

```
10 PRINT CHR$(23)+CHR$(1)
20 x=RND*640:y=RND*400
30 x1=RND*640:y1=RND*400
40 MOVE x,y
50 DRAW x1,y1
60 DRAW x,y
70 GOTO 20
```

That's it for another month !! In the programs section is another example of XOR and we'll probably use it again soon. Until next month, have fun and keep experimenting and if you have any tips or problems you wish to share, please send them in.

Pattern Makers

Both of the programs below utilise the XOR ink mode which has already been explained to achieve the pattern effects. The first program just demonstrates this, with an eye-wrenching pattern.

Time Warp

```
 10 MODE 2
 20 PRINT CHR$(23)+CHR$(1)
 30 FOR i=1 TO 640 STEP 2
 40 MOVE i,400:DRAW 640-i,0
 50 NEXT i
 60 FOR i=1 TO 400 STEP 2
 70 MOVE 1,i:DRAW 640,400-i
 80 NEXT i
 90 FOR i=1 TO 2000:NEXT i
100 GOTO 30
```

The second program also uses XOR but also shows how a pattern can be made using a circle plotting routine. Many names were suggested for this program, ranging from 'Cucumber' to 'Radar Scan', but in the end I decided to call it Explosion and leave it upto you to decide what it really is ! Good luck !

Explosion

```
 10 PRINT CHR$(23)+CHR$(1)
 20 DEG:MODE 2
 30 FOR i=1 TO 360
 40 MOVE 320,200
 50 DRAW 320+200*SIN(i),200+200*COS(i)
 60 NEXT i
 70 FOR i=1 TO 360
 80 MOVE 320,200
 90 DRAW 320+320*SIN(i),200+320*COS(i)
100 NEXT i
```

Some more interesting effects can be achieved by changing the SIN and COS (in lines 50 and 90) around and even substituting TAN for either of them. If you discover any more short, interesting and simple programs, please send them in and we will publish them.

As you will hopefully have read in the section, 'What is my Amstrad?', when you switch on your CPC it is ready to accept commands in BASIC. It tells you this by putting READY on the screen followed by ■. This box is the cursor and shows you where any text that you type in will appear. BASIC is fairly simple to learn as it is similar in many ways to the English language as you will see when we look at the commands. Something to remember is that computers are entirely logical and will only do exactly what they are told to do - nothing more and nothing less.

PRINT - One of the most common BASIC commands is PRINT and it does exactly what it says. Type this into the computer (to get " use shift and at the same time 2) :-

```
PRINT "Hello. I'm your Amstrad."          (Now press ENTER)
```

You should see on the screen :-

```
Hello. I'm your Amstrad.
Ready
■
```

The computer has printed everything inside the quotes and has then told you that it is ready for your next instruction. One important thing to remember is that at the end of each line you must press the ENTER key as from now on we will not remind you. Now type in the following line :-

```
PRINT "Hello.
```

In this line you left out the last quote and so your Amstrad printed all the text until it came to the end of the line. Now try this :-

```
PRINT
```

Here, you haven't told the computer what to print and so it prints a blank line and then tells you it is ready for another instruction. Now try some of your own.

CLS - When you have finished type :- CLS ; this stands for CLear Screen and this is exactly what it does, the screen is cleared of all text. Then you should see Ready in the top left corner of the screen followed by the cursor.

RUN - Until now all the instructions you have typed have been executed as soon as you pressed the ENTER key. However, by giving each line (or command) a LINE NUMBER you can tell the computer to execute them in sequence. Line numbers can start at any number but it is usual for them to start at 10 and to go up in steps of 10 (so you can add extra lines in between if you wish). For example, type in this program :-

```
1Ø CLS
2Ø PRINT "Hello"
3Ø PRINT
4Ø PRINT "I am your Amstrad"
```

This program will clear the screen, print 'Hello', leave a line and then print 'I am your Amstrad' but ONLY WHEN YOU TYPE 'RUN' (note : for commands you may use either upper or lower case letters).

end of the program. When you typed RUN, the computer executed each line of the program in numerical order.

LIST - If you now type LIST you will see the program appear on the screen. As the computer executes a program according to line numbers, if you add,

```
25 PRINT "Friend"
```

after the listing and then LIST the program again you will see that line 25 is now positioned between lines 2Ø and 3Ø. If you run the program you should see :-

```
Hello
Friend

I am your Amstrad
Ready
```

EDIT - If you list the program and type EDIT 2Ø, line 2Ø will appear with a cursor at the beginning of the line. You can move this cursor backwards and forwards along the line by using the cursor keys (these are the keys with arrows on). You can also delete and insert letters by using the CLR, DEL and letter keys. If you change line 2Ø to read :-

```
20 PRINT "Hello ";
```

and then press ENTER and list it you will see that you have changed line 2Ø in the program. Now run the program and this is what you will see :-

```
Hello Friend

I am your Amstrad
Ready
```

The semi-colon (;) that you added to the end of line 2Ø tells the computer to print the next thing immediately after the previous character. If you change the ; in 2Ø to a comma (,) you will find that the word Friend is still printed on the same line but there are several spaces after Hello and before Friend. This is because the , tells the computer to print the next piece of text in the next PRINT ZONE. Each print zone is 13 characters (letters) long.

Here is a brief summary of the commands we have looked at:-

PRINT - Prints text or numbers. CLS - Clears the screen.

RUN - Tells the computer to execute a program.

LIST - Displays a program in the computer's memory.

EDIT - Allows you to change a line in the program.
